Introduction to Machine learning

Agenda

1. ML Introduction
2. AI vs ML vs DL vs DS
3. Simple linear Regression → Mathematical intuition behind it.

2. AI vs ML vs DL vs DS

AI → creating an application where it performs all its task without human intervention.

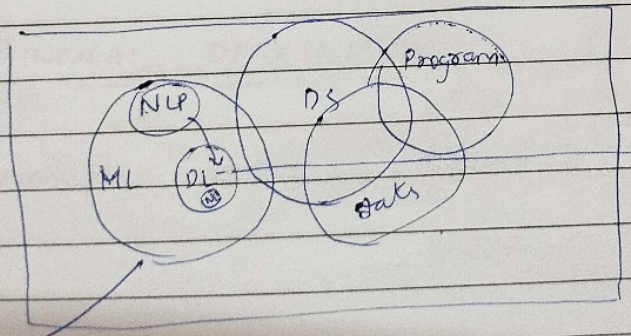
Streaming platform

↓

application → Netflix ← Recommendation

↳ AI model

Eg:- Chatbot, Self driving car, Auto Suggestion, Youtube recommendat



→ DL was created to m
human learning in 19

ML → stats tools to analyze, visualize, perform predictions & other tasks with the help of data.

ML & DL

Supervised

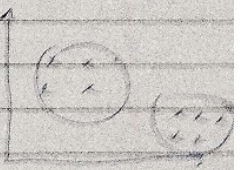
Regression
① Linear Regression
② Multi linear Regression
③ Polynomical
④ SVR
⑤ Decision Tree
⑥ Random Forest
⑦ xg Boost
KNN

Classification
Logistic Regression
SVM
Decision Tree
Naive Bayes
KNN

Unsupervised

Clustering Algorithm
DBSCAN, K-Means, Hierarchical, Mean Clustering

→ In Supervised ML → o/p will be given
→ In Unsupervised ML → o/p is not known.
↓
Here we will make Clusters.



Unsupervised ML

Supervised

Degree	Exp	Salary (Predict) o/p	Feature
B.E	7	90k	Regression Problem
PHD	2	70k	
–	–	–	
–	–	–	

No. of Play hrs	No. of Study hrs	Pass/Fail
9	1	0 → Fail
7	2	0 Fail
3	5	1 Pass

Independent feature, Dependent feature

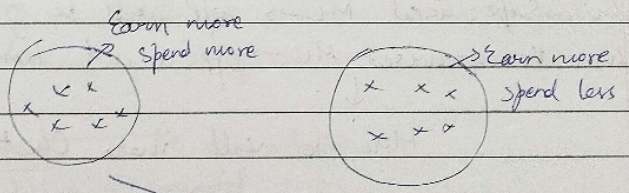
Classification Problem

Eg:- Flight Price Prediction → Regression
Bayesian Forest Fire → Classification
AQI → Regression
Whether tomorrow Rain/Not → Classification
Buying Day of Person → Classification

Unsupervised ML

~~Simple Linear Regression~~

Age	Salary	Spending-score (1-10)	Product
24	70k	1	→ No promotion
26	100k	9	
-	-	(2)	→ 20% promotion
21	20k	9	→ 10% Promotion
25	120k	2	



* This particular scenario is "CUSTOMER SEGMENTATION"

SIMPLE LINEAR REGRESSION

⇒ 1 Independent feature 1 Dependent feature.

Eg:- Dataset

Ht. Wt.

Aim :- I/p → Ht, Predict → Wt.

⇒ Eg: (9) Dataset

No. of Rooms (x) Price (y)

If No. of Rooms ↑ Price ↑.

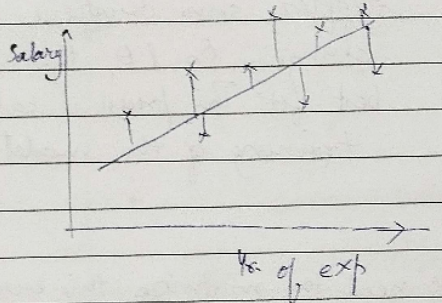
⇒ Eg :- Yr of Exp Salary

Model → will be trained on Yr. of Exp & salary

Predict → Salary based on I/P year

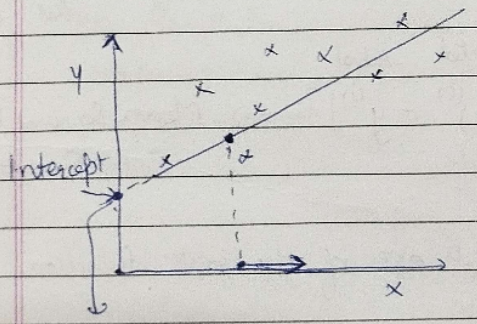
DATASET → Training Dataset

Yr of Exp	Salary
-	-
-	-
-	-
-	-
-	-
-	-



⇒ Main Aim of simple linear Regression is to find the best fit line in such a way that difference b/w real point & predicted point sum will be minimum.

⇒ The diff. b/w ~~real~~ real pt & predicted point → Error/ Residuals



Equation of a straight line

$y = mx + C$

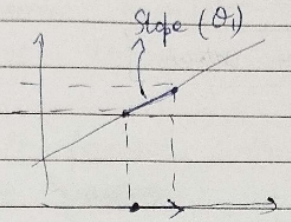
$h_\theta(x) = \theta_0 + \theta_1 x$

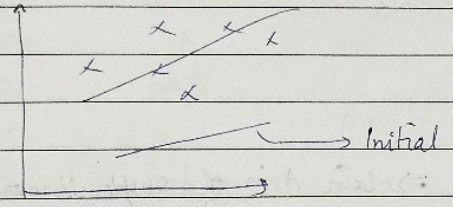
When x value is 0 where is the best fit line meeting y-a this intercept.

Exp	Salary
0	3.25 lakh

θ_1 = slope.

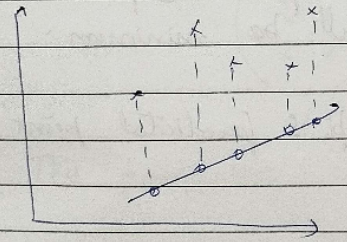
Slope → With the unit movement in x axis what is movement on y axis



⇒ 

θ_0, θ_1 → value

⇒ After some iterations of changing θ_0 & θ_1 the line will best fit. This process is called training of the model.

⇒ 

We need to minimize this error, & we use 'COST FUNCTION'

Cost Function

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right)^2$$

($h_\theta(x)^{(i)}$: Predicted, $y^{(i)}$: Actual)

⇒ Mean Squared Error / Cost Function

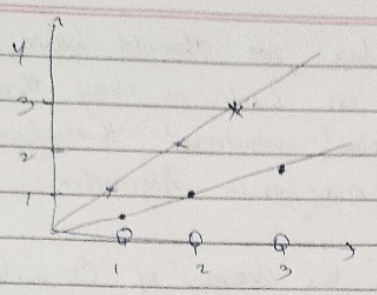
* Mean squared Error is one of the cost function

Final Aim → in linear Regression

$$\text{Minimize, } J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right)^2$$

↓ by changing θ_0 & θ_1 values.

↳ Mean squared error



$\theta_1 = 1, \theta_0 = 0$

Training Data

x	y
1	1
2	2
3	3

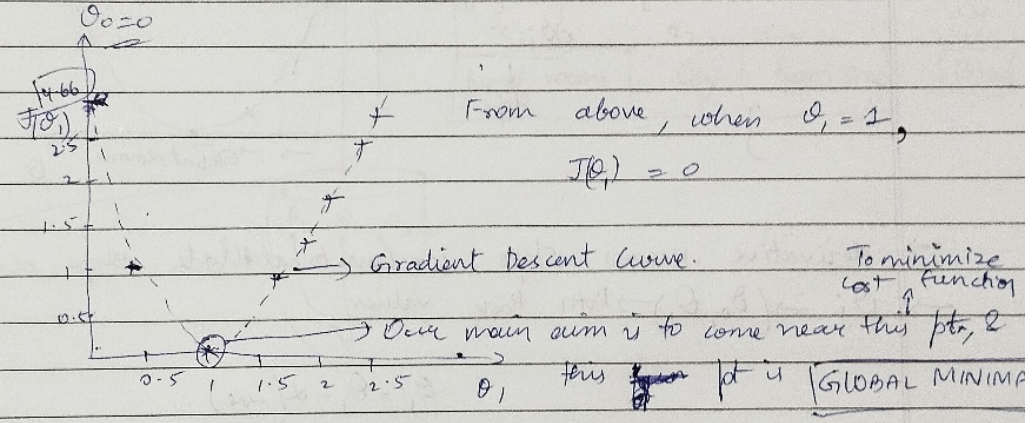
$h_\theta(x) = 1 \quad x = 1$

$h_\theta(x) = 2 \quad x = 2$

$h_\theta(x) = 3 \quad x = 3$

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^i - y^{(i)} \right)^2$$

$$J(\theta_1) = \frac{1}{3} \sum_{i=1}^{3} (0 + 0 + 0) = 0$$



From above, when $\theta_1 = 1$,

$J(\theta_1) = 0$

→ Gradient Descent Curve.

To minimize cost function

Our main aim is to come near this pt, & this pt is GLOBAL MINIMA

→ When $\theta_1 = 0.5$

$h_\theta(x) = 0.5 \quad x = 1$

$h_\theta(x) = 1 \quad x = 2$

$h_\theta(x) = 1.5 \quad x = 3$

→ Plotting the pts. above.

$$J(\theta_1) = \frac{1}{3}\left[(0.5 - 1)^2 + (1 - 2)^2 + (1.5 - 3)^2 \right]$$

$$= \frac{1}{3}\left[0.25 + 1 + 2.25 \right] = \frac{3.5}{3} = 1.1667$$

→ When $\theta_1 = 0$, → Plotting pts. above

$h_\theta(x) = 0$

$$J(\theta_1) = \frac{1}{3}\left[(0 - 1)^2 + (0 - 2)^2 + (0 - 3)^2 \right] = \frac{14}{3} = 4.66$$

There is a problem

* We cannot initialize θ_1 value, we should have a mechanism, it should change in such a way that we should come over 'global minima' therefore for this reason we write Convergence Algorithm.

Convergence Algorithm {Optimize the change of θ_1 value}

↓ Also for θ_0.

Repeat until Convergence

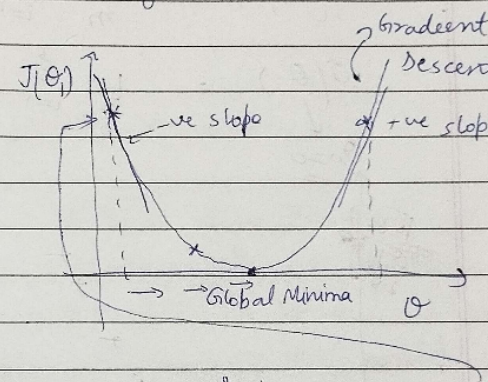
{

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_j)$$

learning Rate

$\frac{\partial}{\partial \theta_j}$ → Slope

}



→ derivative means slope, we have to calculate slope at

→ $\theta_j \rightarrow (\theta_0, \theta_1)$ → Both these values.

$$\theta_j = \theta_j - \alpha(-ve)$$

$$= \theta_j + \alpha \rightarrow \text{Increasing } \theta_j$$

→ When we plot this line downwards we have to increase θ_j to get it reach Global Minima.

→ Slope is used to update ~~i~~ (θ_0, θ_1) value.

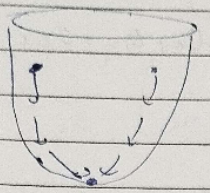
→ In case of that 2nd +ve slope to reach global minima we have to subtract α

$$\theta_j = \theta_j - \alpha(+ve \text{ slope})$$

→ α → (learning Rate) → How fast you want to converge.

Usually selected $\alpha = 0.001$

→ If we include θ_0 also, we will get a 3-D shape.

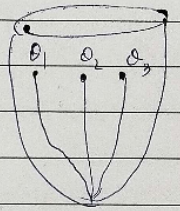


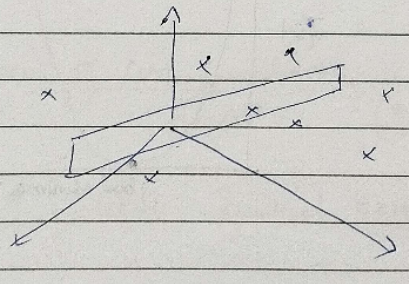
Global Minima

For Multilinear Regression

$$h_\theta(x) = \theta_0 + \theta_1(x_1) + \theta_2 x_2 + \theta_3 x_3$$

x_1	x_2	x_3	y
No. of rooms	City	Room Size	Prize





- In this case instead of best fit line best fine plane will be received.

- More than 3-D it will become hyperplane.

Linear Regression Algorithm

$h_\theta(x) = \theta_0 + \theta_1 x$ $\qquad$ $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_n x_n$

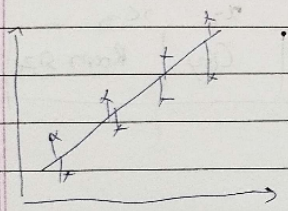
Convergence Algorithm

Cost function,

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right)^2$$ → MEAN SQUARED ERROR

⇒ [Loss function vs Cost function]

↓ One Data point ↓ All observation



Loss function = $\left(h_\theta(x)^{(i)} - y^{(i)}\right)^2 \Rightarrow \left(\hat{y}^{(i)} - y^{(i)}\right)^2$

↓ Predicted ↓ Actual / Truth pt.

CONVERGENCE THEOREM

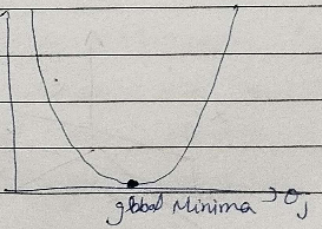
Repeat until Convergence

{

$j = 1$ to m

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_j)$

}



$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_0} \left[\frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right)^2 \right]$$

$h_\theta(x) = \theta_0 + \theta_1 x$

$$J = 0 \qquad = \frac{\partial}{\partial \theta_0} \left[\frac{1}{2m} \sum_{i=1}^{m} \left((\theta_0 + \theta_1 x)^{(i)} - y^{(i)} \right)^2 \right]$$

$$= \frac{1}{m} \sum_{i=1}^{m} \left[(\theta_0 + \theta_1 x) - y^{(i)} \right] \times 1$$

$$= \frac{\partial}{\partial \theta_1} \left[\frac{1}{2m} \sum_{i=1}^{m} \left((\theta_0 + \theta_1 x)^{(i)} - y^{(i)} \right)^2 \right]$$

$$= \frac{1}{m} \sum_{i=1}^{m} \left[(\theta_0 + \theta_1 x)^{(i)} - y^{(i)} \right] \left[x \right]$$

Repeat Until Convergence

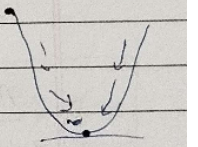
$$\{$$

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right) x^{(i)}$$

$$\}$$

α - Speed of convergence



Global Minim

Types of cost functions.

① MSE ② MAE ③ RMSE

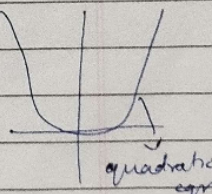
① MSE { Mean Squared Error }

$$MSE = \sum_{i=1}^{n} \frac{(y - \hat{y})^2}{n}$$

↳ quadratic eqn.

$\hat{y} = \theta_0 + \theta_1 x$

↳ Predicted



quadrati eqn

Advantages :

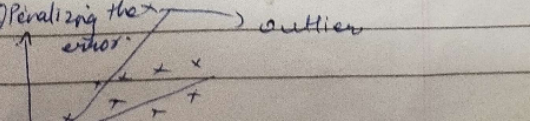
(i) This equation is differentiable

(ii) This equation also has one global minima.

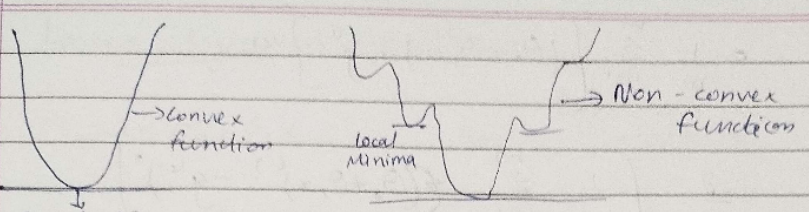
Disadvantage

(1) Not Robust to outliers.

(2) Penalizing the error.



outlier



Exp → Independent

Salary (lakhs) INR → Dependent

$(y - \hat{y})^2$

(Salary - predicted salary)^2 (lakhs)^2

lakhs lakhs

Error → unit is changing

② MAE (Mean Absolute Error)

- $MAE = \frac{1}{n} \sum_{i=1}^{n} |y - \hat{y}|$

Advantage

① Robust to outliers

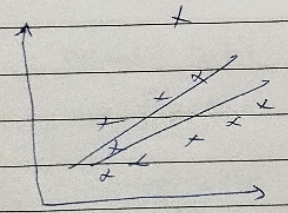
② It will also be same unit

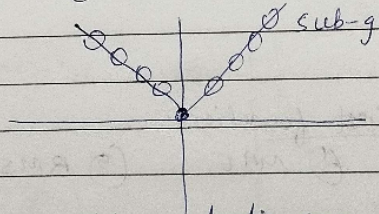
Disadvantage

① Convergence usually takes more time

② Optimization is a complex task.

③ Time consuming.





- Not quadratic eqn (derivative)
- We cannot find out slope when pt is 0.
- Slope is calculated using sub-gradient concept.

Advantages & Disadvantages Unit, Outliers, Differentiable

① Mukes Loss
↓
MSE & MEA

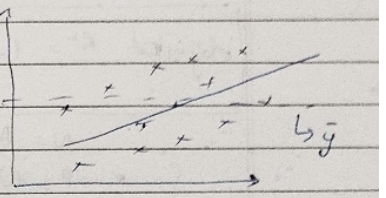
② RMSE
↓
$\sqrt{MSE}$

Performance Matrix

① R Squared ② Adjusted R Squared

① R Squared

$$R\text{squared} = 1 - \frac{SS_{Res}}{SS_{Total}}$$



SS_{Res} = Sum of square Residuals
SS_{Total} = Sum of Square Average

$\bar{y}$ = Avg of y

$$= 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

→ If model is fitted well it will be low
→ high

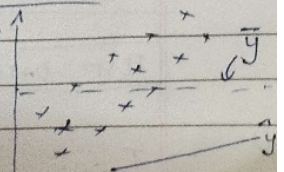
$$R\text{-squared} = 1 - \left\{ \frac{\text{Small number}}{\text{Bigger number}} \right\} \rightarrow \text{small}$$

R-squared /op ≤ 1

↳ 0.85 → 85% Accurate Model
> 0.75 → 75% Accurate

R-Square checks performance of the model created

If R-square is -ve → very bad model.



Adjusted R squared

Size of House	City location	No of bedrooms	Gender	Price

R^2 65% Adj R^2 = 63% P=1

R^2 75% Adj R^2 = 73% P=2

R^2 88% ← Should Not

R^2 90% ← have happened

→ Adj R^2 85%

$$\text{Adjusted } R^2 = 1 - \frac{(1-R^2)(N-1)}{N-P-1}$$

N = No. of data points.

P = No. of Independent features.

Overfitting and Underfitting (Bias and Variance)

Dataset (1000 datapoints) → Training (700) → Model Training

→ Test (300) → Model Test

↓

New Dataset

700

TRAINING DATASET

500 / 200

Training Part | Validation Data

↓ | ↓

Training the model | Hyperparameter tuning the model.

TRAIN DATA	Very Good Accuracy (98%) [Low Bias]
TEST DATA	Very Good Accuracy (85%) [Low variance] → Generalized Model

Training data info is passed by Bias, test data using variance.

Train	Bad Accuracy (High Bias)
Test	Bad Accuracy (High Variance)

Eg ① Train → Very Good Accuracy [90%] → [Low Bias] → Solve this problem by Hyperparameter tuning
Test → Bad Accuracy (50%) → [High Variance]
↓
[Overfitting]

② Train → [High Bias]
Test → Model is low/High → [Low or High Variance]
↓
[Underfitting]